Pitfall in relational database design

A bad design of several properties, including
→ Repitition of information
→ Inability to represent certain information
→ Loss of information

lending schema

branch_name	branch-city	assests	Customer name	loan no.	amount
Downtown	Brooklyn	19,0000	Jones	L-17	1000
Perryridge	Horseneck	2,00,000	Smith	L-23	3000
Redwood	Palo Alto	1,50,000	Jackson	L-29	5000
Downtown	Brooklyn	1,90,000	Jackson	L-14	2000
Perryridge	Horsneck	2,00,000	Glenn	L-16	22000

Repeating of information in our alternative design is undesirable. Repeating information wastes space.

→ It complicates updating the databases.

For ex → the assests of Perryridge branch change from 2,00,000 to 3,00,000. Under our design, many tuples of the lending relation need to be changeds. Thus, updates are most costly. We ensure that every tuple consists to the Perryridge branch is updated, else our database will show two different asset values for the Perryridge branch.

Another problem with lending schema design is that we cannot represent directly the information concerning a branch (branch_name, branch_city, assets) unless there exist atleast one loan at the branch. This is because tuples in the lending relation require values for loan_number, amount and customer_name.

→ One sol^n to this problem is to introduce null values but null values are difficult to handle. If we are not willing to deal with null values, then we can create the branch information only when the first loan app^n at ~~that branch~~ is made. Worse, we would have to delete this information when all the loans have been paid.

Clearly, this situation is undesirable, under our database design, the branch information would be available regardless of whether or not loans are currently maintained in the branch and without resorting to null values.

Decomposition

→ It is the process of breaking down relation into ~~to~~ multiple relation.

→ It should be lossless because it confirms that the information in the original relation can be accurately reconstructed based on the decomposed relations.

→ If there is no proper decomposition of the relation, then it may lead to problems like loss of information.

→ Decomposition helps in eliminating some of the problems of bad design such as redundancy, inconsisties

There are two type of decomposition

→ Lossy → The decomposition of relation R into R1 and R2 is lossy when the join of R1 and R2 doesn't yield the same relation as in R.

One of the disadvantages of decomposition into two or more relational schemes is that some information is lost during retrieval of original relation or table.

Consider that we have student with three attribute roll_no., s_name, department.

student

Roll_no	s_name	department
111	parimal	Computer
222	parimal	Electrical

This relation is decomposed into two relation no-name and name-department

no_name

Roll_no.	Sname
111	Parimal
222	Parimal

name_department

Sname	dept
parimal	comp.
parimal	Electrical

In lossy decomposition, spurious tuples are generated when a natural join is applied to the relations in the decomposition.

stu_joined

Roll_no.	Sname	dept
111	parimal	Comp.
111	parimal	Electrical
222	"	Comp.
222	"	Electrical

The above decomposition is a bad decomposition or lossy decomposition.

Losless join decomposition

The decomposition of relation R into R1 and R2 is losless when the join of R1 and R2 yield the same relation in R.

The lossless - join decomposition is always defined with respect to a specific set of dependecies.

stu-name

Roll_no.	s_name
111	parimal
222	parimal

stu-dept

Roll_no.	Dept
111	Comp.
222	Electrical

stu-joined :

Roll_no.	s_name	dept
111	parimal	computer
222	parimal	Electrical

In lossless decomposition, no any spurious tuples are generated when a natural join is applied to the relation in the decomposition

Properties of decomposition

→ Lossless decomposition → deco
→ Dependency preservation
→ Lack of Data Redundancy →

1) It gives a gurantee that the join will result in the same relation as it was decomposed.

2) Dependency Preservation → Every dependency must be satisfied by alleast one decomposed table. If $(A \rightarrow B)$ holds, then two sets are functional dependent. And it become more useful for checking the dependency easily if both sets in a same relation.

$R(A, B, C, D, E)$

$R_1(A, B, C)$ $\qquad$ $R_2(C, D, E)$

$A \rightarrow B$ $\quad$ $B \rightarrow C$ $\qquad$ $AD \rightarrow E$ ✗

3) Lack of Data Redundancy → The proper decomposition should not suffer from any data redundancy. Lack of data redundancy is also known as a repitition of information.

Functional dependency

It is a relationship that exists when one attribute uniquely determines another attribute.

⇒ If R is a relation with attribute X and Y.

X	Y
a	1
a	2
b	1
b	3

$X \rightarrow Y$

Y is functionally dependent on X.
Here X unquidly determines the Y value.

⇒ It is a set of constraints b/w two attributes in a relation.

X → determinant or identification key

Y dependant

eg → If every attribute B of R dependent of A, then attribute A is a primary key.

$A \rightarrow B$

ID	Name	Surname
S_1	Bhanu	P
S_2	Priya	G
S_3	Bhanu	M

{ ID → Name }
{ ID → Surname }

Functional dependencies can be categorized two types —

Trivial	Non-trivial
$AB \rightarrow A$ (dependency is valid)	$\beta \not\subseteq \alpha$
if $\beta \subseteq A$	$AB \rightarrow ABC$

Closure set of attributes :

Attribute closure of an attribute set 'A' can be defined as a set of attributes which can be functionally determined from it. Denoted by F^+.

or

$R(A, B, C, D)$

$A \rightarrow B$

$B \rightarrow D$

$C \rightarrow DE$

$CD \rightarrow AB$

→ The set of all those attributes which can be functionally determined from an attribute set is called as a closure of that attribute set.

→ Closure of attribute set (X) is denoted by $(X)^+$

$A^+ = \{A B D\}$

$B^+ = \{B D\}$

$C^+ = \{C D E A B\}$

$D^+ = \{D\}$

$E^+ = \{E\}$

$(AB)^+ = \{A, B, D\}$

$(AD)^+ = \{A, D, B\}$

$(ABD)^+ = \{A B D\}$

step-1 Add the attributes contained in the attribute set for which closure is being calculated to the result set.

step-2 Recursively add the attributes to the result set which can be functionally determined from a.s. already contained in the result set

~~Q → R(X, Y, Z, W) is decomposed into~~

Amstrong's Axioms

i) Union Rules

If $A \to B$ holds and $A \to C$ holds, then $A \to BC$ holds.

$A \to B$
$A \to C$

$A \to BC$

ii) Decomposition Rule

If $A \to BC$ holds then $A \to B$ and $A \to C$ holds

iii) Augmentation Rule

R (A, B, C)

If $A \to B$ holds and Y is attribute set, then AY ~~and~~ BY also holds.

iv) Transitivity

if $A \to B$
$B \to C$
then
$A \to C$

(vi) Reflexivity rule

If X is a set of attributes and Y is subset of X, then we would say $X \to Y$.
(trivial FD)

v) Pseudotransitivity ↓

If $A \to B$ holds
and $BC \to D$ holds

$AC \to D$ holds

Minimal functional dependency set ↓
(Irreducible set of FD)

Minimal cover → It means to eliminate any kind of redundancy from a FD set.

$R(WXYZ)$

$X \to W$

$WZ \to XY$

$Y \to WXZ$

> $\alpha \to \beta$
>
> उलट α side something extra and β side something extra and worst case full dependency extra.

In step-1
apply decomposition rule ↓

$X \to W$

$WZ \to X$

$WZ \to Y$

$Y \to W$

$Y \to X$

$Y \to Z$

Now, case is FD may be redundant. We don't check right hand side attribute because now we write seperate here.

$X^+ = XW$

again we compute X^+ without seeing $X \to W$. If we get again XW without seeing this means $X \to W$ is redundant.

$X^+ = X$

I am not getting W here. So $X \to W$ is essential.

(2) $WZ \to X$

$(WZ)^+ = (WZXY)$

again

$(WZ)^+ = (WZYX)$

so $WZ \to X$ is redundant.

(3) $WZ \to Y$

$(WZ)^+ = WZYX$

again

$(WZ)^+ = WZ$

so $WZ \to Y$ is essential.

(4) $Y \to W$

$Y^+ = YWXZ$

again

$Y^+ = YXZW$

so $Y \to W$ is redundant.

(5) $Y \rightarrow X$

$Y^+ = YXWZ$

again

$Y^+ = YZ$

so $Y \rightarrow X$ is essential.

(6) $Y \rightarrow Z$

$Y^+ = YXW$

so $Y \rightarrow Z$ is essential.

After that

$X \rightarrow W$

$WZ \rightarrow Y$

$Y \rightarrow X$

$Y \rightarrow Z$

But again problem arises. Is there any redundancy on left hand side or not. In this ex. $WZ \rightarrow Y$

$(WZ)^+ = WZYX$

again

$W^+ = W$

$Z^+ = Z$

so $WZ \rightarrow Y$ is essential.

Finding the keys Using Closure

Super key → If the closure result of an attribute set contains all the attributes of the relation, then that attribute set is called as a superkey of that relation.

In above example

→ The closure of attribute A is the entire relational schema.

→ Thus, attribute A is a super key for that relation.

Candidate key

If there exists no subset of an attribute set whose closure contains all the attributes of the relation, then that attribute set is called as a candidate key of that relation.

Example

→ No subset of attribute A contains all the attributes of the relation.

→ Thus, attribute A is also a C.K. for that relation.

How to find out a candidate key?

i) R(A B C D E F G H)

$AB \to C$

$BD \to EF$

$AD \to G$

$A \to H$

$(ABD)^+ = (ABDCEFGH)$

↓

candidate key

ii) R(A B C D E)

$AB \to CD$

$D \to A$

$BC \to DE$

$(B)^+ = B$

$(AB)^+ = (ABCDE)$

$(BC)^+ = (ABCDE)$

$(BD)^+ = BDACE$

C.K.

$(BE)^+ = BE$ ✗

iii) R(A B C D E)

$BC \to ADE$

$D \to B$

$(C)^+ = ?$

$(AC)^+ = AC$ ✗

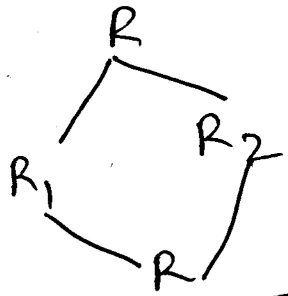
$(BC)^+ = (BCADE)$ ✓

$(CD)^+ = (CD)$

$= (CDBAE)$ ✓

$(CE)^+ = $ ✗

Loseless join decomposition



This property guarantees that the extra or less tuple generation problem doesn't occur after decomposition.

R

A	B	C	D
1	a	p	x
2	b	q	y
~~3~~	~~a~~	α	

R_1		R_2
A	B	D
1	a	x
2	b	y

R_1

A	B
1	a
2	b

R_2

C	D
p	x
q	y

$R_1 \times R_2$

A	B	C	D
1	a	p	x
1	a	q	y
2	b	p	x

→ If a relation R is decomposed into two relations R_1 & R_2, then it will be lossless-if

i) $attr(R_1) \cup attr(R_2) = attr(R)$

ii) $attr(R_1) \cap attr(R_2) \neq \phi$

iii)

R

A	B	C
1	a	p
2	b	q
3	a	r

R_1

A	B
1	a
2	b
3	a

R_2

B	C
a	p
b	q
a	r

$R_1 \bowtie R_2$

A	B	C
1	a	p
2	b	q
3	a	r
1	a	r
3	a	p

~~3tate~~ if I make common attribute candidate key then it will never generate extra tuples.

(iii) $attr(R_1) \cap attr(R_2) \rightarrow attr(R_1)$ (key proof)

or

$attr(R_1) \cap attr(R_2) \rightarrow attr(R_2)$

example of loseless decomposition

A	B	C	D	E
a	122	1	p	w
b	234	2	q	x
a	568	1	r	y
c	347	2	s	z

1 condition violate
$R_1 (A, B) \cup R_2 (C, D) = R$

(1) $R_1 (A, B)$, $R_2 (C, D) \rightarrow$ Lossy

(2) $R_1 (ABC)$, $R_2 (D, E)$

i) $R_1 \cup R_2 = A, B, C, D, E$

(ii) $R_1 \cap R_2 \neq \phi$
$(A, B, C) \cap (D, E)$ violate // Lossy

(3) $R_1 (A, B, C)$, $R_2 (C, D, E)$

i) $R_1 \cup R_2 = A, B, C, D, E$

(ii) $R_1 \cap R_2 \neq \phi$
"C"

(iii) violate | above table 1 is repeated

(4) $R_1 (A, B, C, D)$, $R_2 (A, C, D, E)$

$R_1 \cup R_2 = ABCDE$

$R_1 \cap R_2 = \{A, C, D\}$

$R_1 \cap R_2 \Rightarrow$ attri (R_1)

Dependency Preserving

If a table R having FD set F, is decomposed into two tables R_1 and R_2 having FD set F_1 and F_2

$$F_1 \subseteq F^+$$

$$F_2 \subseteq F^+$$

$$\overline{(F_1 \cup F_2)^+ = (F)^+}$$

$R(A, B, C)$

$$F: \begin{bmatrix} A \rightarrow B \\ B \rightarrow C \\ C \rightarrow A \end{bmatrix}$$

F_1 $\quad \underline{R_1(A, B)} \mid \underline{R_2(B, C)}$ $\quad F_2$

$$\begin{tikzcd}
 & R(F) \arrow[dl, dash] \arrow[dr, dash] & \\
R(F_1) \arrow[dr, dash] & & R(F_2) \arrow[dl, dash] \\
 & R(F_1 \cup F_2) = F &
\end{tikzcd}$$

$A^+ = AB$

$B^+ = BCA$

$\left[\begin{array}{l} A \to B \checkmark \\ B \to A \end{array}\right]$

$R_2(B, C)$

$B^+ = B\,\cancel{C}\,A$

$\left[\begin{array}{l} B \to C \\ B \to A \end{array}\right]$

$C^+ = CAB$

$\left[\begin{array}{l} C \to A \\ C \to B \end{array}\right]$ Dependency preserving

---

$R(A, B, C, D)$

$AB \to CD$

$D \to A$

F_1: $R_1(A, D)$

$A^+ = A$

$D^+ = DA$ $D \to A$

$\cancel{(AD)^+ = AD}$ ×

F_2: $R_2(B, C, D)$

$B^+ = B$

$C^+ = C$

$D^+ = DA$ $D \to A$ ×

$(BC)^+ = BC$ ×

$(BD)^+ = BDAC$

$BD \to A$ ×

$BD \to C$ ✓

$(CD)^+ = CDA$ × $CD \to A$ ×

Prime attribute

Attributes that form a candidate key of a relation i.e. attributes of candidate key are called prime attributes and rest of the attributes of the relation are non-prime attributes.

Partial FD ↓

Partial dependency means that a non-prime attributes is functionally dependent on a part of a candidate key.

For ex → R(A, B, C, D)

$A \rightarrow B$

$D \rightarrow C$

$(AD)^+ = (ADBC)$ ~~is~~

AD is a candidate key.

B & C are Non prime attribute

A & D are prime attribute

so here partial FD exists

Full FD ↓

When a non-prime attributes is fully functional dependent on the candidate key.

R(A, B, C, D)

$ABC \rightarrow D$

D is Non Prime attribute

ABC are prime attributes.

So Full FD.

"Normalization"

→ Normalization is the process of decomposing a big relation into smaller relation.

→ The prime objective of normalization is to reduce redundancy.

Redundancy leads the problem of inconsistency.

Goal of Normalization — (Requirement)

To achive

i) Functional dependency preservation

2) loss less join decomposition

Idea → In the table studentinfo we have tried to store entire data about student.

Result → Entire branch data of a brach must be repeated for every student of the branch.

Redundancy → When same data is stored multiple times unnecessarily in a database

Disadvantages → (i) Insertion, deletion and modification anomalies
(ii) Inconsistency (data)
(iii) Increase in database size and increase in time (slow)

Student info

S_id	name	age	Br_code	Br_name	HOD_name
1	A	18	101	CS	XYZ
2	B	19	101	CS	XYZ
3	C	18	101	CS	XYZ
4	D	21	102	EC	PQR
5	E	20	102	EC	PQR
6	F	19	103	ME	KLM

Insertion anomalies: → when certain data (attribute) cannot be inserted into DataBase, without the presence of other data

Deletion anomalies - If we delete some data (unwanted), it cause deletion of some other data (wanted)

Updation/Modification anomalies - when we want to update a single piece of data, but it must be done at done

Student info

S_id	name	age	Br code	Br name	HoD_name
1	A	18	101	CS	XYZ
2	B	19	101	CS	XYZ
3	C	18	101	CS	XYZ
4	D	21	102	EC	PQR
5	E	20	102	EC	PQR
6	F	19	103	ME	KLM

student_info

S_id	name	age	Br code
1	A	18	101
2	B	19	101
3	C	18	101
4	D	21	102
5	E	20	102
6	F	19	103

Branch_info

Br code	Br name	HoD name
101	C.S	XYZ
102	E.C	PQR
103	M.E	KLM

→ As one paragraph contains a single idea similary one table must contain direct & main data about an Entity

→ Normalization (Decomposition of tables) of table is done of the basic of functional dependen

First Normal Form ↓

→ It is a theoretical discussion.

→ Every cell can contain atomic value.

or

you can say that you don't have multivalued attribute.

→ Every relation is in 1NF if ~~it doesn't~~ every cell contain atomic value.

→ domain should be same (every column)

Roll No.	name	Course
101	Modi	CN OS
102	Sonia	DBMS CO

⇓

Roll No.	name	Course
101	Modi	CN
101	Modi	OS
102	Sonia	DBMS
102	Sonia	CO

Second Normal Form ↓

For a table to be in the Second Normal form, it must satisfy two conditions—

1) The table should be in the 1NF.
2) There should be no Partial Dependency.

$R(A, B, C, D)$

$AB \rightarrow D$
$B \rightarrow C$

$(AB)^+$ is a candidate key

$A, B \in$ prime attributes
$C, D \in$ non-prime attributes

$B \rightarrow C$ is a partial dependency

When the non-prime attribute is depend on the part of a candidate key.

So it is not in 2NF.

We have to convert in 2NF.

$R(ABCD)$

$R_1(ABD)$ ⇓ AB is C.K.

$R_2(BC)$ B is a C.K.

How to identify it?

It is in 2NF.

Q → R (A B C D E)

$AB \to C$ (P.D.)

$D \to E$ (P.D.)

R (A B C D E)

$(ABD)^+ = ABDCE$

So $(ABD)^+$ is a candidate key.

So it is not in 2NF.

R (A B C D E)

- R_1 (A B C)
- R_2 (D E)
- R_3 (A B D)

More detailed discussion on 2NF ⇃

R(A B C)

$B \rightarrow C$

so $(AB)^+$ is a candidate key

$A, B \in$ prime attribute

$C \in$ Non Prime attribute

so it has partial dependency.

R(A B C) → R_1(A B), R_2(B C)

A	B	C
a	1	x
b	2	y
a	3	z
c	3	z
d	3	z
e	3	z

R1

A	B
a	1
b	2
a	3
c	3
d	3
e	3

R2

B	C
1	x
2	y
3	z
3	z
3	z
3	z

3NF →

Transitive Dependency → A F.D. from $\alpha \to \beta$ is called transitive if $\alpha, \beta \in$ non-prime

3NF →

A relation is in 3NF if

a) It is in 2NF

b) No transitive dependency

---

every dependency from $\alpha \to \beta$

i) either α is super key

ii) or β is a prime attribute

R(A, B, C)

$A \to B$

$B \to C$

A is a candidate key

- R_1 (B, C)
- R_2 (A B)

A	B	C
a	1	x
b	1	x
c	1	x
d	2	y
e	2	y
f	3	z
g	3	z

R_2

A	B
a	1
b	1
c	1
d	2
e	2
f	3
g	3

R_1

A	C
1	x
2	y
3	z

Q→ R (A B C D E)

A → B

B → E

C → D

$(AC)^+ = ABCED$

So $(AC)^+$ is candidate key

A → B is P.D.

C → D is P.D.

B → E (transitivity)

- R_1 (A B E)
- R_2 (C D)
- R_3 (A C)
- R_4.

(3) R(A B C D E)

$A \to B$

$B \to E$

$C \to D$

$(AC)^+$ is a C.K.

B, D, E Non-Prime Attribute

A B C D E

- $R_1(A B E)$
  - $R_{11}(A, B)$
  - $R_{12}(B E)$
- $R_2(C D)$
- $R_3(A C)$

Q→ R(A B C D E F G H I J)

$AB \to C$

$A \to DE$

$B \to F$

$F \to GH$

$D \to IJ$

- $R_1(A D E I J)$
  - $R_{11}(A D E)$
  - $R_{12}(D I J)$
- $R_2(B F G H)$
  - $R_{21}(B F)$
  - $R_2(F G H)$
- $R(A B C)$

(AB) is a candidate key.

Q→ R(A B C D E)

$AB \rightarrow C$

$B \rightarrow D$

$D \rightarrow E$

$(AB)^+$ is a candidate key.

- $R_1(B\ D\ E)$
  - $R_{11}(B\ D)$
  - $R_{12}(D\ E)$
- $R_2(A\ B\ C)$

BCNF

$\alpha \rightarrow \beta$

↓

superkey

R(A B C)

$AB \rightarrow C$

$C \rightarrow B$

(AB), (AC) is a c.k.

It is 3NF not BCNF.

Identify the Normal Form :-

① R (A B C D E F G H)

AB → C

A → DE

B → F

F → GH

(AB) C.K.

1NF

② R(A B C D E)

CE → D

D → B

C → A

(CE) is C.K.

1NF

Q R(ABCDE)

AB → CD

D → A

BC → DE

(AB) (BD) (BC)

3NF

③ R(A B C D E F)

AB → C

DC → AE

E → F

(ABD) (BCD) C.K.

1NF

Q→ R(ABCDE)

BC → ADE

D → B

(BC) (CD)

3NF

④ R(A B C D E G H I)

AB → C

BD → EF

AD → GH

A → I

1NF

(ABD) is C.K

BCNF

• Example (BCNF) :-

R (A B C)

$AB \rightarrow C$

$C \rightarrow B$

A	B	C
a	1	x
b	2	y
c	2	z
c	3	w
d	3	w
e	3	w

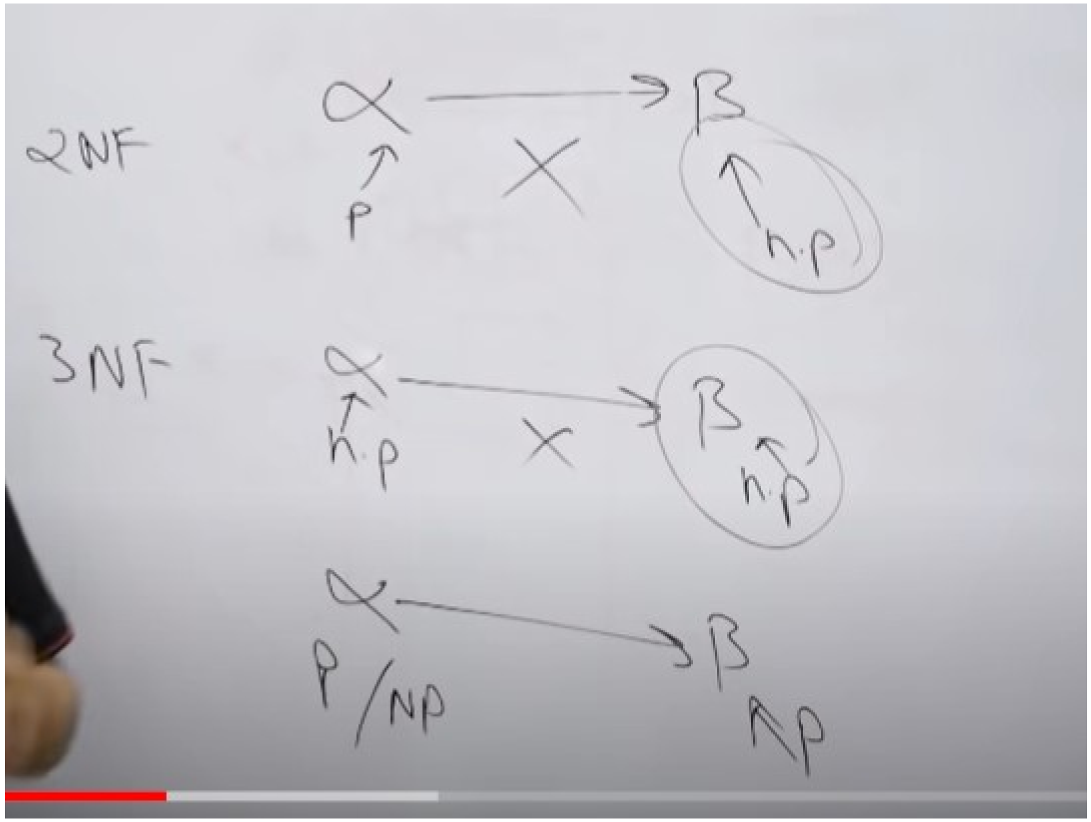
R_1 (CB)

R_2 (AC)

A	C
a	x
b	y
c	z
c	w
d	w
e	w

CB

C	B
x	1
y	2
z	2
w	3


2NF
α → β
p
n.p
3NF
α → β
n.p
n.p
α → β
p/np

BCNF (Boycee Codd Normal Form)

A Relation R is said to be in BCNF if every FD, $X \rightarrow A$ in F satisfy At least one of following condition hold.

1) $X \rightarrow A$ must be trival
2) X is a super key.

Note :- 1) In BCNF the FD may not be preserv
2) BCNF is more stricker then 3NF.

BCNF

- Example (BCNF) :-

R (A B C)

$AB \to C$

$C \to B$

A	B	C
a	1	x
b	2	y
c	2	z
c	3	w
d	3	w
e	3	w

R_1 (C B)

R_2 (A C)

A	C
a	x
b	y
c	z
c	w
d	w
e	w

C B

C	B
x	1
y	2
z	2
w	3

Fourth Normal Form (4NF)

A relation 'R' is in 4NF if and only if the following condition are satisfied —

i) if 'R' is in 3NF or BCNF

ii) if it contains no MVD's

<u>Now</u> What is Multivalued Dependency

→ For a dependency $x \rightarrow y$, if for a single value of x, multiple value of y exists, then the relation have multivalued dependency.

→ The relation should have atleast three attributes $(x \rightarrow\rightarrow y)$; $(x \rightarrow\rightarrow z)$

→ The attributes y and z should be independent of each other.

Note [F.D $(\alpha \rightarrow \beta)$ says we can't have two tuples with same α value but different β value].

Consider Multivalued Dependency

E_id	ProjectName	D_Name
101	Java/DBMS	Sita/Khushi
102	OS/CN	Geeta/Ram

E_id $\twoheadrightarrow$ ProjectName

E_id $\twoheadrightarrow$ D_Name

employee-id "multideterminés" D_Name

In this ex - ProjectName and D_Name both are independent attributes and it is dependent on e-id.

ex→ Consider the database table of a class which has two relations R1 contains studentID (SID) and student name (SNAME) and R2 contains course id (CID) and course name (CNAME)

R1 (SID, SNAME)

SID	SNAME
S1	A
S2	B

R2 (CID, CNAME)

CID	CNAME
C1	C
C2	D

Table R1XR2

SID	SNAME	CID	CNAME
S1	A	C1	C
S1	A	C2	D
S2	B	C1	C
S2	B	C2	D

Multivalued dependencies (MVD) are-

SID →→ CID ; SID →→ CNAME ;

SNAME →→ CNAME

Note → MVD occurs if two or more independent relations are kept in a single relation

Note – 4NF is a level of a database normalization where there are no non-trivial multivalued dependencies other than a candidate key.

How to decompose it in 4NF?

E_id	ProjectName	D_Name
101	Java/DBMS	Sita/Khushi
102	OS/CN	Geeta/Ram

Eid	Project Name	D_Name
101	Java	Sita
101	DBMS	Sita
101	Java	Khushi
101	DBMS	Khushi
102	OS	Geeta
102	CN	Geeta
102	OS	Ram
102	CN	Ram

R1

Eid	ProjectName
101	Java
101	DBMS
102	OS
102	CN

R2

E_id	D_Name
101	Sita
101	Khushi
102	Geeta
102	Ram

5th NF or Project Join Normal Form

→ 5NF is rarely used practically but it is useful for theoritical study.

→ 5NF is based on Join dependency.

→ Join Dependency
– Decompose the relation in multiple relations and it should be loseless and maintain dependencies of original relation.

→ A relation is in 5NF –
i) It must be in 4NF
ii) No Join Dependency exists

for ex →

Dept	subject	student
CSE	DBMS	Shreya
IT	CN	Yug
CSE	OS	Geeta
CSE	COA	Sita
ME	APP	Rini
EC	CSA	Ira

Dept →→ subject
Dept →→ student

Dept	subject
CSE	DBMS
IT	CN
CSE	OS
CSE	COA
ME	APP
EC	CSA

Dept	student
CSE	Shreya
IT	Yug
CSE	Geeta
CSE	Sita
ME	Rini
EC	Ira

If we perform join here

Dept	subject	student
CSE	DBMS	Shreya ✓
CSE	DBMS	Geeta ✗
CSE	DBMS	Sita ✗
IT	CN	Yug ✓
CSE	OS	Shreya
CSE	OS	Geeta ✓
CSE	OS	Sita ✗
CSE	COA	Shreya ✗
CSE	COA	Geeta ✗
CSE	COA	Sita ✓
ME	APP	Rini ✓
EC	CSE	Isa ✓

so we have to decompose it in 3 Table

Dept	subject

dept	student

Subject	Student